VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Mechanizmy riadenia robotického auta NXP

Driving Mechanisms of Robotic Car NXP

2020 Frederik Zajac

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Frederik Zajac**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Mechanizmy řízení robotického auta NXP
Driving Mechanisms of Robotic Car NXP

Jazyk vypracování: slovenština

Zásady pro vypracování:

Cílem práce je vytvořit software pro ovládání robotického auta NXP model „Alamak" s vývojovým kitem FRDM-K64. Software umožní účast na soutěži NXP Cup a ve vybraných dodatečných disciplínách. Konkrétně v disciplínách vyhnutí se překážce a nouzové brzdění.

Software umožní:
1. Detekci dráhy a okrajových čar.
2. Připojení a využití ultrazvukového senzoru pro detekci překážek.
3. Dynamické přizpůsobení světelným podmínkám dráhy.
4. Řízení auta po závodní trati.
5. Splnění vybraných dodatečných disciplín ze závodů NXP. A to konkrétně vyhnutí se překážce a nouzové brzdění.

Práce bude obsahovat:
1. Přehled používaných metod a algoritmů.
2. Implementaci výše popsané funkcionality.
3. Experimenty a vyhodnocení výsledků.
4. Dokumentaci programového řešení s využitím diagramů jazyka UML.

Seznam doporučené odborné literatury:

[1] Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides (Gang of Four): Návrh programů pomocí vzorů. Grada. Praha 2003. ISBN 8024703025
[2] DARWIN, Ian F. Java cookbook. 2nd ed. Sebastopol, CA: O'Reilly, c2004, xxiv, 829 p. ISBN 05-960-0701-9. Dostupné z: http://it-ebooks.info/book/2249/

Dále dle pokynů vedoucího práce.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **Ing. David Ježek, Ph.D.**

Datum zadání: 01.09.2019

Datum odevzdání: 30.04.2020

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 14. mája 2020

.....

Súhlasím so zverejnením tejto diplomovej práce podľa požiadavkou čl. 26, odst. 9 Študijného a skúšobného rádu pre štúdium v bakalárskych programoch.

V Ostrave 14. mája 2020

.....

Rád by som sa na tomto mieste poďakoval vedúcemu práce Ing. Davidovi Ježkovi, Ph.D. za ochotu a trpezlivosť pri zasvätení do problematiky hardwaru a vypracovaní tejto bakalárskej práce. Rovnako by som sa chcel poďakovať celej svojej rodine a priateľke, za podporu.

Abstrakt

Táto bakalárska práca sa zameriava na vylepšenie doterajšieho modelu riadenia autonómneho auta, ktoré sa zúčastní celosvetovo známej súťaži NXP Cup. Pomocou pridaných ultrazvukových senzorov sme schopný prijímať dáta, vďaka ktorým dokážeme analyzovať okolie robotického auta. Mojou úlohou je implementovať vhodný a efektívny algoritmus, ktorý pomocou dát dokáže prepočítavať vzdialenosť rôznych prekážok, ktorým je nutné sa vyhnúť a zároveň držať sa na trati medzi vodiacimi čiarami. Taktiež sa v práci popisuje mechanizmus núdzového brzdenia v prípade neočakávanej prekážky.

Kľúčové slová: NXP Cup, Autonómne vozidlo, FRDM-K66F, HC-SRO4

Abstract

This bachelor thesis focuses on improving the current model of driving a robotic car, which will participate in the world famous competition NXP Cup. Using the added ultrasonic sensors, we are able to receive data, thanks to which we are able to analyze the surroundings of the robotic car. My task is to implement a suitable and efficient algorithm that can calculate the distance of various obstacles to be avoided while keeping the track between the guide lines using data. The work also describes the emergency braking mechanism in case of unexpected obstacles.

Keywords: NXP Cup, Autonomous vehicle, FRDM-K66F, HC-SRO4

Obsah

Zoznam použitých skratiek a symbolov

CSV	–	Comma-separated Values
GPIO	–	General-Purpose Input/Output
I/O	–	Input/Output
IRQ	–	Interrupt ReQuest
LED	–	Light Emitting Diode
MCU	–	Microcontroller Unit
MPU	–	Microprocessor Unit
PIT	–	Periodic Interrupt Timer
RAM	–	Random Access Memory
SRAM	–	Static Random Access Memory
TFC	–	The Freescale Cup
UDP	–	User Datagram Protocol
USB	–	Universal Serial Bus
Wi-Fi	–	Wireless Fidelity

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov zdrojového kódu

1 Úvod

O autonómnych autách sa hovorí v posledných rokoch mnoho, ale prvé myšlienky a testy s podobnými konceptami tu už boli pred mnohými rokmi. Autonómne riadenie by malo byť zárukou niekoľkých výhod, ako napríklad zvýšená bezpečnosť, či zníženie nehodovosti. V dnešnej dobe však vieme, že aj najmenšia chyba autopilota môže spôsobiť hotové nešťastie. Ako napredujú technológie, autonómne riadenie sa stáva súčasťou nášho bežného života v rôznych oboroch. Najviac voľnosti dostávajú vo výrobných fabrikách, kde sa rôzne roboty pohybujú na základe senzorov a vodiacich čiar úplne samé. V automobilovom priemysle si však takéto niečo nemôžeme úplne dovoliť. Hoci sú už dnešné autopiloty vcelku dokonalé, napriek tomu sa vyžaduje osobná prítomnosť a pozornosť za volantom, aby sme sa vyhli nepredvídateľným situáciám.

V tejto práci sa budem snažiť priblížiť už vytvorený autonómny model autíčka a jeho fungovanie. Tento model sa zúčastnil minulý rok súťaže NXP Cup a dosiahol veľmi dobré výsledky v disciplínach, aj napriek tomu, že neboli použité rôzne ultrazvukové senzory, alebo senzory na zisťovanie rýchlosti autíčka. Mojou úlohou je teda nadviazať na túto prácu a vylepšiť ju v rámci možností tak, aby dané disciplíny zvládalo čo najlepšie.

Z dôvodu momentálnej svetovej situácie a ochodenia COVID-19, mi nebolo umožnené dostať sa cez hranice a nemohol som pracovať na reálnom modeli autíčka. Z tohto dôvodu som sa dohodol s vedúcim bakalárskej práce Ing. Davidom Ježkom, Ph.D na tom, že bude možné v domácich podmienkach nasimulovať fungovanie ultrazvukových senzorov, pomocou ktorých som sa mal zamerať najmä na disciplíny vyhýbania sa prekážkam a núdzového brzdenia.

Jednou z hlavných úloh je vymyslieť osadenie a následne funkčnosť ultrazvukových senzorov, aby bolo možné čo najpresnejšie merať vzdialenosti objektov pred senzormi. Následne bolo potrebné sa zamerať na funkčnú časť a stavbu domáceho modelu autíčka tak, aby bolo možne čo najpresnejšie nasimulovať situácie, ktoré by boli podobné tím, ktoré by sa objavovali pri testoch v školskom laboratóriu.

Táto práca opisuje celkové správanie senzorov v rôznych situáciách, na ktorej konci sa budem snažiť vyjadriť svoj osobný názor na vzniknutú prácu, ale aj ideu možnej funkčnosti v budúcnosti.

2 NXP Cup súťaž

Súťaž NXP Cup je celosvetová súťaž s mnohoročnou tradíciou. Zúčastňujú sa jej stredoškolskí a vysokoškolskí študenti s tým rozdielom, že súťažia na iných dráhach. Súťažiaci svojou účasťou získavajú cenné skúsenosti, ktoré potom môžu uplatniť v praxi. Autonómne riadenie vozidiel dnes považujú najväčšie svetové automobilky za dôležitý smer vývoja a budúcnosť automobilovej dopravy.

Počas vypracovania práce na ktorú sa nadviazalo, sa zmenili pravidlá. Zmena pravidiel sa týka šírky dráhy, ktorá bola zmenšená z 60cm na 55cm a zároveň je oficiálna dráha tvorená novým materiálom. Ďalej je povolené využiť ľubovoľné MCU alebo MPU (prípadne kombináciu oboch). Čo sa týka vývojových dosiek, všetky musia byť vytvorené firmou NXP, alebo musia obsahovať MCU / MPU firmy NXP. Pri autách s podvozkami typu Alamak aj Model C sa súťaží v rovnakej kategórii. Informácie o súťaži a pravidlách v tejto kapitole boli čerpané z [1].

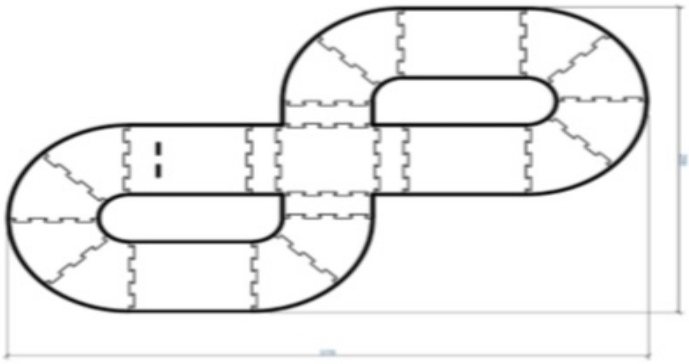
2.1 Disciplíny

Pre všetkých účastníkov je povinný hlavný závod , v ktorom sa dá získať najvyšší počet bodov a 4 dodatočné disciplíny, na získanie extra bodov. Tímy závodia na rovnakom rozložení pretekárskych tratí bez ohľadu na model auta. Pri závode sa hodnotí čas a auto nesmie vyjsť z dráhy viac než dvoma kolesami. Dráha je do začatia závodu utajená, aby si účastníci nemohli dráhu vopred vyskúšať.

Obr. 1: Ukážka dráh [2]

2.1.1 Dráha v tvare 8

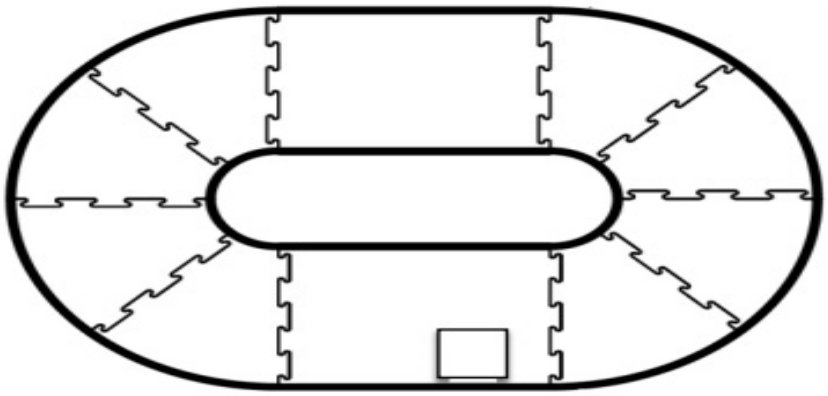
Ako už vyplýva z názvu, trať bude v tvare osmičky a závodníci majú presne 90 sekúnd na to, aby prešli čo najviac kôl. Počítajú sa len dokončené kolá. Dokopy majú 3 pokusy a medzi každým pokusom je čas 30 sekúnd na vylepšenie, respektíve prepnutie auta do iného stavu, v ktorom by mohlo autíčko zvládnuť túto trať lepšie.



Obr. 2: Ukážka dráhy v tvare 8

2.1.2 Vyhnutie sa prekážkam

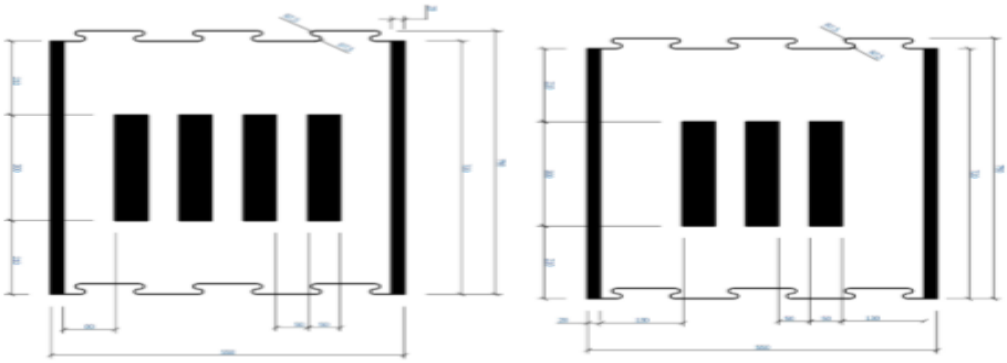
Na túto disciplínu je vyhradených 90 sekúnd. V tejto disciplíne sa na trať položí prekážka v tvare kocky s rozmermi 20x20x20cm. Je povolený len jediný pokus a pokus sa ráta iba vtedy, ak sa kolesá autíčka, alebo časť kolies nedostanú za hraničné čiary. V tomto prípade sa do finálneho bodovania neráta rýchlosť autíčka, takže je povolená rýchlosť podľa vlastného uváženia.



Obr. 3: Ukážka dráhy s prekážkou

2.1.3 Zóna obmedzenia rýchlosti

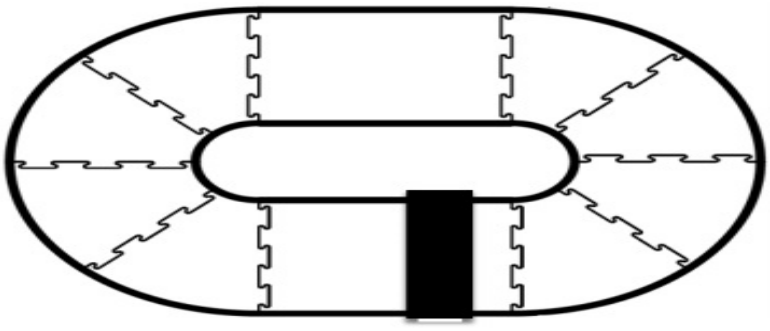
Táto disciplína spočíva na zvyšovaní a znižovaní rýchlosti na trati v tvare oválu. Autíčko bude umiestnené tak, aby štartovalo pred troma pásmi, kde má značne zvýšiť svoju rýchlosť. Keď dorazí k zóne spomalenia, ktorá je vyznačená štyrmi pruhmi, autíčko musí výrazne znížiť svoju rýchlosť a to o približne polovicu pôvodnej rýchlosti. Následne, ak opäť príde do zóny s troma pásmi, musí zrýchliť na svoju pôvodnú rýchlosť. Disciplína končí, ak auto splní všetky tieto podmienky.



Obr. 4: Ukážka dráhy so zónami rýchlosti

2.1.4 Núdzové brzdenie

Disciplína núdzového brzdenia spočíva v tom, aby vozidlo zastavilo pred prekážkou, bez toho aby sa jej dotklo. Autíčko je umiestnené na určitú časť trati a musí nabrať rýchlosť. Následne spraví 2 kolá a potom sa na trať položí prekážka z polystyrénu alebo podobného materiálu bielej farby s rozmermi 20 x 20 x 60cm. Táto prekážka bude cez celú šírku trate a autíčko musí zastaviť pred touto prekážkou, teda nemôže ju nijak obísť. Na túto disciplínu je povolený len jeden pokus a body sú udelené, ak autíčko splní vyššie spomenuté podmienky.



Obr. 5: Ukážka dráhy s núdzovým brzdením

3 Autonómny model auta

V tejto kapitole by som sa rád vyjadril a zhrnul autonómny model auta, ktorý vytvoril Richard Zvonek a na ktorý nadväzuje moja práca, v ktorej sa pokúsim vylepšiť už vytvorený model autíčka. Hlavnou úlohou mojej práce je pridať ultrazvukové senzory na model auta tak, aby sme mohli v súťaži NXP Cup konkurovať v ďalšej disciplíne, ktorá spočíva vo vyhýbaní sa objektom na dráhe, prípadne núdzovo zabrzdiť, ak nie je možné sa vyhnúť prekážke tak, aby sme nevyšli z jazdnej dráhy. Väčšina informácií v tejto kapitole boli čerpané z [3].

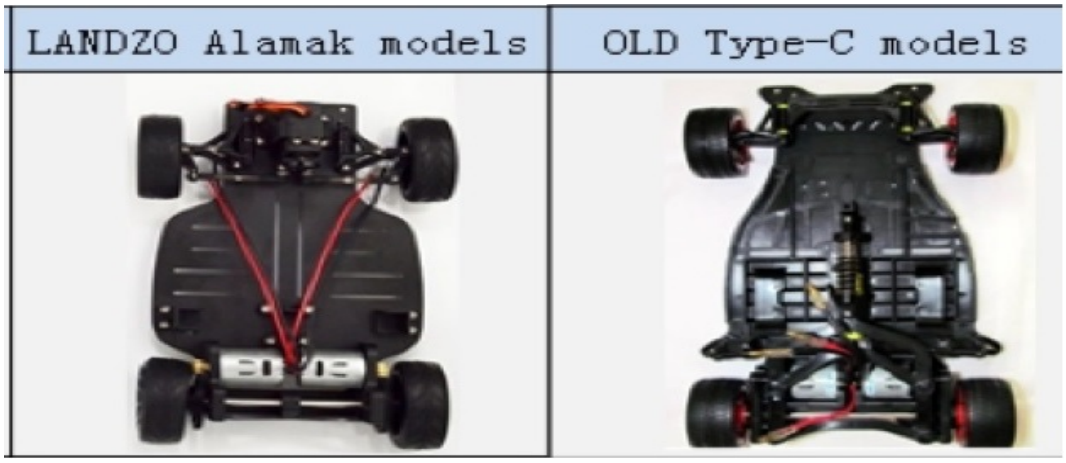
3.1 Popis hardwaru

3.1.1 Podvozok

Na model auta je možné vybrať si z dvoch podvozkov. Prvým a starším typom podvozku je Type-C, ktorý má na rozdiel od novšieho podvozku LANDZO Alamak segmentové telo, je o niečo menší a má slabšie motory. Autor projektu si zvolil pre model auta, ktoré sa priamo zúčastnilo súťaže NXP Cup podvozok Alamak, napriek tomu však používal aj starší typ podvozku na domáce testovanie. Celkové porovnanie je na obrázku 6. [4]

Tabuľka 1: Porovnanie podvozkov Alamak a Type-C

Model	Alamak	Type-C
Telo	Jeden kus	Segmenty
Veľkosť	28,5x16x7cm	28,5x16x8cm
Motory	2x 7,2V 380	2x 7,2V 260
Priemer pneumatík	35mm	50mm
Rázvor kolies	16cm	16cm



Obr. 6: Porovnanie podvozkov

3.1.2 Vývojové dosky

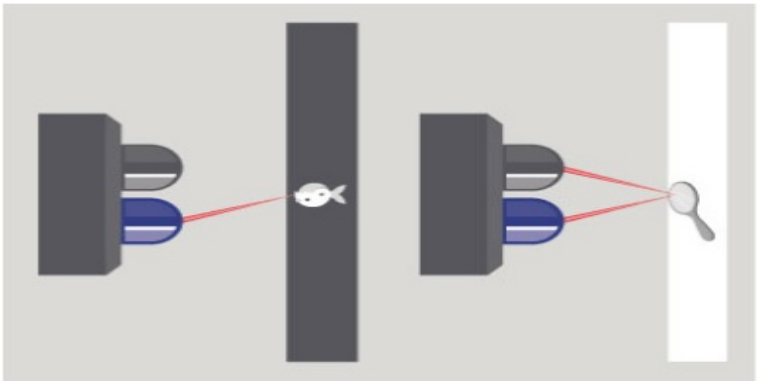
Ako bolo vyššie spomenuté, vývoj autíčka prebiehal v dvoch verziách. Domáca verzia so starším podvozkom typu C obsahovala vývojovú dosku FRDM-KL25Z, ktorá oproti súťažnej verzií s vývojovou doskou FRDM-K66F zaostáva najmä po výkonnostnej stránke. K doske FRDM-KL25Z bola pomocou USB rozhrania pripojená ešte vývojová doska Raspberry PI 3 B, ktorá disponovala väčším výkonom a bolo teda umožnené rýchlejšie spracovanie obrazu. Ďalej obsahuje vstavané Wi-Fi zohranie, či dokonca umožňuje prístup pomocou SSH. K obom doskám boli pripojené TFC shieldy, čo sú vlastne rozširujúce dosky pre kompatibilitu s periférami.

Tabuľka 2: Porovnanie dosiek FRDM-KL25Z, Raspberry Pi 3 B a FRDM-K66F [5] [6] [7]

Vlastnosti	FRDM-KL25Z	Pi 3 B	FRDM-K66F
CPU	MKL25Z128VLK4	Broadcom BCM2837	MK66FN2M0VMD18
Počet jadier	1	4	1
Takt procesoru	48 MHz	1,2 GHz	180 MHz
SRAM	16KB	-	256 KB
RAM	-	1GB	-
Flash pamäť	128 KB	-	2 MB

3.1.3 Infračervený senzor

Podmienkou súťaže je zastavenie autíčka po prejazde cieľovou čiarou. Kvôli tomu sa na model autíčka musel osadiť infračervený senzor, ktorého hlavnou funkciou je detekcia telies, meranie otáčok kolesa, či prechod farby z čiernej na bielu. V tomto prípade sa využívala práve posledná spomenutá možnosť, pretože infračervené svetlo je odrážané na svetlých povrchoch a pohlcované na tmavých povrchoch. Tento senzor vyžaruje infračervené svetlo pomocou infračervenej LED a následne čaká, kým sa svetlo vráti na fototranzistor. Pokiaľ sa svetlo odrazí, teda je namierené oproti svetlému povrchu, vracia 1. V opačnom prípade dostávame 0. [8]



Obr. 7: Princíp fungovania IR senzoru

3.2 Popis softwaru

3.2.1 Podporný software

V rámci vývoja bolo potrebné vytvoriť podporný systém, vďaka ktorému by bolo možné zobrazovať dáta, ktoré sú zbierané zo senzorov na autíčku. Pre tieto účely Richard Zvonek vyvinul aplikáciu NXP Car Interface, ktorá v reálnom čase zobrazuje dáta z kamery. Kvôli veľkému množstvu dát ich však nebolo možné prenášať v upravenom stave, preto sa posielajú v takzvanej surovej forme a následne sú upravované rovnakými algoritmami, ktoré sa používajú na spracovanie aj mikroprocesorom autíčka. Aplikácia teda zobrazuje originálny obraz z kamery, na ktorý sa následne aplikujú rôzne filtre, ktorý daný obraz spracúvajú. Aplikácia je taktiež schopná ukladať dáta z jazdy do CSV, kde sa ukladajú všetky numerické hodnoty, ktoré sú rovnako zobrazované aj v aplikácii.

3.2.2 Prenos dát

Na bezdrôtový prenos dát medzi počítačom a autíčkom bol použitý UDP protokol, ktorý zaisťuje asynchrónny a spojovo orientovaný prenos. Vývojová doska je v tomto prípade UDP server a aplikácia NXP Car Interface je klientom. Na prenos dát je použitá trieda SendData, ktorá prenáša nespracované dáta z kamery.

3.2.3 Získavanie dát z riadkovej kamery

Na modeli autíčka sa nachádza riadková kamera, ktorá musí byť vhodne umiestnená na modeli autíčka. Ideálne je to v určitej výške nad autíčkom, aby sme dokázali snímať z dostatočnej výšky obraz pár centimetrov pred autíčkom. Kamera zbiera dáta ktoré sa následne musia spracovávať, čo poznáme pod pojmom spracovanie obrazu.

Vo väčšine prípadov sa nám pod pojmom spracovania obrazu vybaví spracovanie dát dvojrozmerných polí, respektíve matíc. V našom prípade však dostávame dáta z riadkovej kamery, čo už naznačuje, že nepôjde o klasické dvojrozmerné pole, ale len jednorozmerné pole. Na vytvorenie skutočného obrazu teda potrebujeme tieto riadky skladať a tým vytvoríme 2D obraz. Následne prebiehajú ďalšie fázy spracovania obrazu, ako sú:

1. Filtrovanie mediánom

    - Filtrovanie mediánom sa používa na elimináciu šumu z obrazu. Mediánová filtrácia je veľmi efektívna na odstránenie šumu. Algoritmus medianového filtra je v princípe veľmi jednoduchý. Iterujeme obrazom a každý iterovaný bod si uložíme aj s jeho okolím. Vytvorí sa matica rozmeroch N x N, kde každý bod nadobúda určitú hodnotu jasu a N je klasicky v rozsahu 3 až 5 pixelov. Tieto hodnoty sa usporiadajú a vyberie sa prostredná hodnota, čo je vlastne medián. Táto hodnota sa následne dosadí do iterovaného bodu. [9]

(a) Ukážka aplikácie mediánoveho filtra

5	6	10
15	**55**	8
25	4	11

5	6	10
15	**10**	8
25	4	11

4, 5, 6, 8, **10**, 11, 15, 55.

(b) Ukážka matice na výpočet mediánu

Obr. 8: Ukážka princípu mediánového filtra

2. Normalizácia obrazu

   - Normalizácia obrazu rieši problém so zlými svetelnými podmienkami. V spracovaní obrazu je normalizácia proces, ktorý mení rozsah hodnôt intenzity pixelov. Zmyslom tejto normalizácie je dosiahnuť konzistentnosť výsledného obrazu, teda eliminovať oblasti s horšími svetelnými podmienkami. Preto je potrebné nájsť hodnotu minimálneho a maximálneho bodu v obraze a následne tieto hodnoty normalizovať do intervalu <0,255>. Keďže hodnoty z kamery sú v intervaly $< 0, 4095 >$, je potrebné tieto hodnoty konvertovať na nižšie, čo môže spôsobiť zhoršenie obrazu. To sa však na testoch nepreukázalo ako chyba. [10]

3. Prahovanie priemerom

   - Prahovanie priemerom je proces spracovania obrazu, ktorý je známy aj pod pojmom binarizácia, kde obraz prevádzame do binárneho tvaru, teda do tvaru 0 alebo 1, pričom 0 je braná ako čierna farba a 1 je braná ako biela farba. To znamená, že ak je hodnota pixelu menšia ako prahová hodnota, zmeníme danú hodnotu na 0. Ak je hodnota pixela väčšia ako prahová hodnota, skonvertujte ju na 1. Ako prvé však potrebujeme zistiť prahovú hodnotu. Tá sa vypočíta ako priemerná hodnota bodov v obraze. [11]

Aby sme však dostávali správne dáta z riadkovej kamery, musíme zabezpečiť aby bola kamera správne nastavená. Ako je spomenuté vyššie, kamera musí byť osadená v určitej výške, pod nejakým uhlom. Ďalším faktorom je aj zaostrenie kamery. Na kalibráciu kamery bola teda vytvorená kalibračná doska, ktorá sa používa aj v disciplíne zóny obmedzenia rýchlosti. Ako prvé sa nastaví ostrenie kamery tak, aby boli viditeľné všetky čiary na doske. Následne sa

Obr. 9: Ukážka aplikácie prahovania priemerom [11]

autíčko postaví tak, aby sa predné kolesa dotýkali okraja dosky na vyznačených miestach a aby boli rovnobežné s čiarami na doske. Potom už len stačí natáčať kamerou tak, aby bola chyba zobrazená v aplikácii NXP Car Interface čo najnižšia.

3.2.4 Vyhľadávanie čiar

Hľadanie čiar je jedna z najdôležitejších funkcií autíčka k tomu, aby bolo schopné sa udržať na trati bez toho, aby z nej vybočil. Kvôli tejto problematike bol autor tohto autíčka nútený vymyslieť zopár algoritmov, ktoré by čo najefektívnejšie riešili túto problematiku.

Testami sa ukázalo, že najvhodnejším riešením bol takzvaný algoritmus na vyhľadávanie regiónov. Tento algoritmus uchováva body okrajov rovnakej farby a aj danú farbu. Následne sa regióny hľadajú v cykle tak, že sa uložená farba porovnáva s farbou nasledujúceho bodu. Pokiaľ sa tieto farby nezhodujú, indikuje to koniec regiónu a začiatok nového regiónu, pričom sa aktualizuje aj daná farba.

Problémy však nastávajú v zatáčkach, alebo križovatkách, kedy kamera nenájde ohraničenie bieleho regiónu čiernymi. Táto situácia nastáva vtedy, keď je autíčko v zatáčke a riadková kamera nevidí vnútorný čierny región. Tento problém bol vyriešený tak, že autíčko sa odsunie od vonkajšieho regiónu k vnútornejšiemu. Ukážka autíčka, ktoré nabieha do zatáčky, v ktorej nie je vidieť vnútorný okraj trate je zobrazený na obrázku 10.

Aj keď je algoritmus hľadania regiónov pomerne efektívny, vo väčšine prípadov ako sú rovné plochy bez zatáčok, je zbytočne zložitý. Preto bol vyvinutý takzvaný podporný algoritmus, ktorý neprechádza celý obraz. Tento algoritmus hľadá v aktuálnom snímku čierne regióny a porovnáva ich s predchádzajúcimi snímkami. Ak teda algoritmus zistí, že čierne regióny sa nachádzajú na rovnakých miestach ako v predošlom snímku, nepotrebuje využívať zložitejší algoritmus vyhľadávania regiónov.

Obr. 10: Autíčko v zatáčke, kde vidí len jednu čiernu čiaru [12]

3.2.5 Správanie autíčka v zatáčke

Pri jazde autíčka na trati je dôležité mať určené rýchlosti, ktoré sa aplikujú na určité časti trate. Keďže autíčko má dva motory, kde každý pracuje pre jedno náhonové koleso, môžeme s týmito rýchlosťami pracovať. To znamená, že ak sa autíčko rúti po rovine, oba motory budú pracovať rovnako a autíčko môže ísť aj rýchlejšie.

Problém nastáva v zatáčkach, kedy autíčko nielenže musí spomaliť oboma motormi, ale musí pre každé koleso nastaviť inú rýchlosť. To znamená, že ak máme napríklad pravotočivú zatáčku, ľavé koleso sa musí točiť rýchlejšie ako pravé, pretože musí prejsť väčšiu trasu. Zároveň však celková rýchlosť musí byť nastavená na takú hodnotu, aby autíčko nemohlo v zatáčke opustiť jazdnú dráhu.

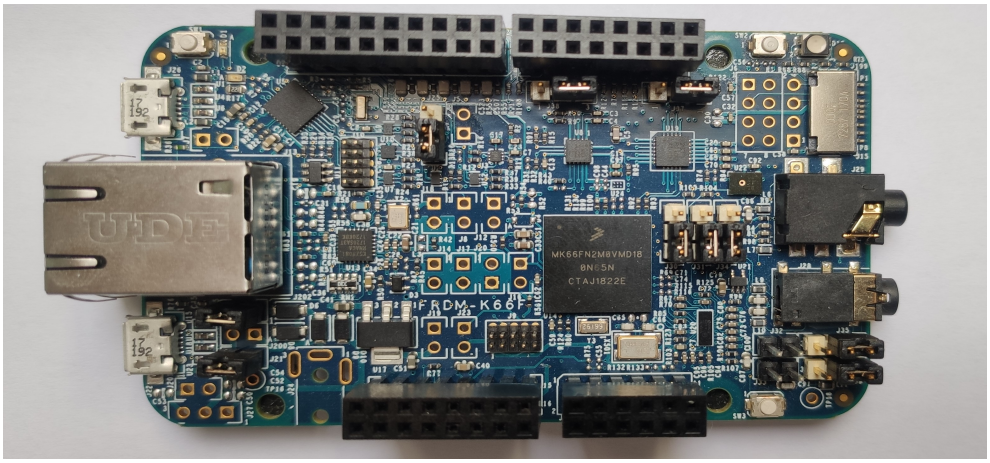
Aby sme mohli rozpoznávať či je auto v zatáčke alebo nie, je používaný mechanizmus historizáciu obrazu, ktorá si uchováva v históriu spracovaných regiónov a pre oba čierne regióny je vypočítaný medián. Pokiaľ sa teda okraje aktuálneho regiónu nachádzajú v blízkosti mediánu, autíčko je s najväčšou pravdepodobnosťou na rovine a teda oba motory môžu fungovať na plný výkon. Ak sa však jeden, alebo oba okraje nenachádzajú v blízkosti mediánu, autíčko je pravdepodobne v zatáčke a k tomu sa musí prispôsobiť aj rýchlosť.

Taktiež je potrebné vziať do úvahy, že autíčko môže do zatáčky prísť v rôznej rýchlosti. Táto rýchlosť závisí od dĺžky rovnej trate, ktorú prešlo. Môže teda nastať situácia, kedy autíčko pôjde po dlhej rovnej trati, teda naberie svoju maximálnu rýchlosť. V tomto prípade musíme do zatáčky spomaliť výraznejšie, ako keby autíčko prešlo po kratšej trase, na ktorej nestihlo nabrať svoju maximálnu rýchlosť.

4 Popis a práca s doskou FRDM-K66F

Vývojová doska FRDM-K66F ktorá pochádza z rodiny K66, je nízko nákladová doska rady Kinetis, ktorá obsahuje sadu softwarových a hardwarových nástrojov na vývoj. Poskytuje taktiež spätnú kompatibilitu s už existujúcim členmi Kinetis rodiny. Rovnako je zaistená aj vyššia periférna integrácia s Dual USB a 10/100 Mbit/s Ethernet pripojením. Hardware tejto dosky je jednoduchý, sofistikovaný a ponúka tieto základné vlastnosti: [5]

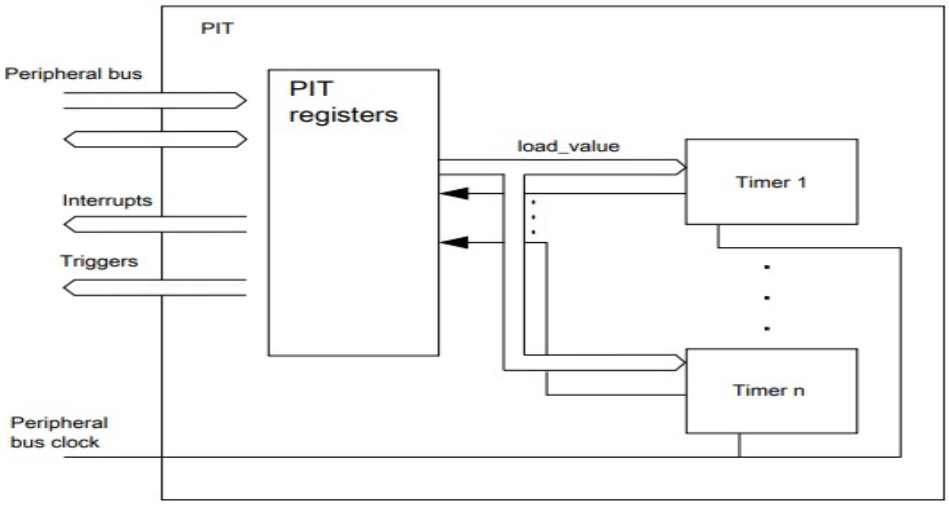
1. Výkon
    - obsahuje jadro ARM Cortex-M4, ktoré obsahuje procesor MK66FN2M0VMD18 s maximálnym výkonom na úrovni 180 MHz.
    - až 32-kanálový DMA pre periférne a pamäťové služby so zníženým zaťažením procesora a rýchlejšou priepustnosťou systému s asynchrónnou podporou v režime zastavenia.
2. Pamäť
    - ponúka 256 kB RAM a pamäť programu o veľkosti až 2MB s vysokou spoľahlivosťou, ktorá zaručuje štvorstupňovú bezpečnostnú ochranu.
3. Konektivita
    - duálne vysokorýchlostné rozhranie USB
    - ethernetový kontroler
4. Systém a časovanie
    - Štyri časovače periodického prerušenia - PIT
    - dva 16-bitové moduly PWM s časovým spínačom a nízkym príkonom
5. Analógové a digitálne IO
    - 54 pinov pripojených k doske s napätím 3,3 V
6. Komponenty na doske
    - FXOS8700CQ - Akcelerometer a magnetometer
    - FXAS21002 - Gyroskop
    - 2 užívateľské tlačidlá
    - 1 RGB LED



Obr. 11: Doska FRDM-K66F

4.1 Časovač periodického prerušenia - PIT

Na prepočítavanie vzdialenosti objektov a analýzu okolia auta potrebujeme v pravidelných intervaloch prepočítavať vzdialenosti, ktoré prijmanie pomocou ultrazvukových senzorov. Na prepočet týchto vzdialeností však potrebujeme počítať čas, za ktorý prejde zaslaná zvuková vlna z vysielača k prekážke a odtiaľ späť na prijímač. Na toto počítanie používame takzvaný IRQ handler, ktorý nám periodicky inkrementuje lokálnu časovú premennú. Pomocou nej vieme následne zisťovať čas a prepočítať sa k výslednej vzdialenosti. Aby sme však na toto počítanie času nemuseli používať prerušovanie na úrovni procesoru, využívame modul PIT, ktorý generuje presné prerušenia v pravidelných intervaloch s minimálnym zásahom do procesora. PIT obsahuje rad časovačov, ktoré vyvolávajú prerušenia a spúšťajú DMA kanály. Modul môže spustiť DMA prenos až na štyroch samostatných kanáloch.



Obr. 12: Diagram PIT [13]

4.2 Ultrazvukový senzor HC-SRO4

Jedná sa o pomerne lacný ultrazvukový senzor, s rozmermi 45x20x15mm, ktorý sa často používa pri vývoji na Arduino doskách, na meranie vzdialeností s pomerne veľkou presnosťou. Samotné zariadenie je schopné merať s presnosťou až 3mm a jeho rozsah merania je od 2cm až do 4 metrov pri 15° uhle snímania, pričom od vzdialenosti 2 metrov stráca na svojej presnosti. Tento senzor obsahuje vysielač ultrazvukových vĺn, prijímač, obvody ktoré sa starajú o chod snímača a 4 piny, ktoré zabezpečujú napájanie senzora s 5V, I/O a uzemnenie.

4.2.1 Funkčnosť

Funkčnosť senzoru je riadená pomocou vstupného a výstupného pinu. Cez vstupný (TRIGGER) pin posielame napätie 5V, čím senzor vysiela ultrazvukové vlny na pracovnej frekvencií 40Hz. Podľa dokumentácie by mal TRIGGER signál mať dĺžku 10 µs, čím zašle zvukové vlny, na ktoré už čaká prijímač. Po prijatí signálu sa na výstupnom (ECHO) pine zobrazí 1, čo značí prijatie.

Veľmi často sa stretávame s problémom, že senzor prijíma takzvané falošné signály, ktoré môžu prísť z rôznych odrazených objektov a hrán. Tento nežiadúci efekt sa dá eliminovať tým, že počkáme určitý čas na prijatie signálu. Keďže sa zvuk šíri rýchlosťou 340m/s pri teplote vzduchu 20°, veľmi jednoducho sa dá vypočítať, za aký čas by nám mali signály doraziť, ak dosah snímača je 4 metre. Potrebný čas vypočítame podľa nasledovného vzorca:

$$\frac{1}{340} \cdot 8 \cdot 10^3 \approx 24ms \tag{1}$$

Podľa tohto vzorca, by nemalo dochádzať prijímaniu falošných signálov po uplynutí 24ms, napriek tomu sa v dokumentácií tohto senzoru odporúča doba až 60m/s. Počas testovania tohto senzoru som sa však nestretol so žiadnymi problémami pri meraní, takže vypočítaná hodnota je podľa môjho názoru dostačujúca, vzhľadom na to, že na konštrukcií modelu autíčka budú umiestnené dva senzory. [14]

4.2.2 Fungovanie 2 senzorov

Aby bolo zabezpečené správne fungovanie dvoch senzorov v reálnom čase, bolo nutné nastaviť fungovanie snímačov tak, aby nepracovali v rovnakej dobe. To by viedlo k veľkému chaosu pri prijímaní signálov a nevedeli by sme, či daný signál patrí danému senzoru. Z toho dôvodu sa pri meraní tieto senzory striedajú, čo znamená, na začiatku cyklu je zvolený jeden, ktorým sa začne a čaká sa, kým nepríjme signál, alebo neuplynie doba 24ms. Následne je na rade druhý senzor a celé sa to opakuje.

4.3 Algoritmus merania

Predtým, ako som sa pustil do práce a programovania na samotnej doske FRDM K66-F, vyskúšal som si fungovanie ultrazvukového senzora na platforme Arduino, k čomu v dnešnej dobe nájdete na internete plno zdrojov a ukážok celého fungovania.

Na vytvorenie testovacieho algoritmu bol použitý nový projekt, ktorý sa neskôr len implementuje to skutočného projektu autíčka. V tomto projekte bolo potrebné predtým, ako sme začali využívať niektoré piny na doske, ich zadefinovať v časti vyhradene na obsluhu pinov pre danú dosku. V tejto časti sa dá vybrať z celej rady pinov, ktoré sú dostupné a taktiež je možné im zadefinovať určité funkcie. Pre prácu s ultrazvukovými senzormi boli vybrané 4 GPIOA piny a pre LED kontrolky boli vyhradené 2 GPIOA piny . Každý ultrazvukový senzor potrebuje jeden vstupný a jeden výstupný pin. Táto časť je pomerne mylná a spôsobila na začiatku komplikácie. Problém spočíva v tom, že v tejto časti sú piny brané opačne, respektíve z pohľadu vývojovej dosky. To znamená, že ak chceme do vstupného pinu TRIGGER poslať napätie, respektíve logickú 1, musí táto hodnota prejsť outputom dosky, čo znamená, že tento pin zadefinujeme ako OUTPUT. Rovnaký princíp platí aj pre druhý pin. Ak chceme z výstupného pinu dostať dáta, musí prejsť inputom do dosky na danom pine. Preto ECHO pin je nastavený na input a zároveň je tento pin nastavený tak, aby nastalo prerušenie pri každej nábežnej hrane, ktoré sa následne v kóde dá obslúžiť.

Spomínaný kód, ktorého hlavná časť je ukázaná nižšie na Výpise 1, je vlastne algoritmus celého fungovania. Funkčnosť tohto kódu som okrajovo rozobral v kapitole 4.2.1, kde je spomínaný celkový princíp fungovania a pracovania so senzormi. V skratke potrebujeme periodicky zasielať zvukové vlny a následne čakať na ich prijatie, čím je možné vypočítať vzdialenosť, za pomoci rýchlosti šírenia zvuku vo vzduchu.

V časti výpisu je možné si všimnúť sekciu extern, ktorá je vyhradená priamo pre funkcie obsluhujúce PIT. Funkcia PIT0_IRQHandler je nastavená tak, aby sa periodicky spúšťala každých 10ms a vo vnútri funkcie inkrementujeme globálnu premennú, ktorá je neskôr používaná na určenie uplynutého času.

Funkcia PORTA_IRQHandler slúži na detekovanie prerušenia. Na začiatku sa do lokálnej premennej načíta hodnota pinu, na ktorej bolo zaznamenané prerušenie. Táto hodnota je v desiatkovej sústave a predstavuje bitový posun logickej jednotky o číslo, ktoré je priradené danému pinu. Ako je z kódu vidieť, táto funkcia počíta za pomoci nameraného času vzdialenosť, ktorá je kontrolovaná a pri určitej hodnote sa dá vyvolať rozsvietenie LED kontroliek. Princíp je jednoduchý, pri prečítaní logickej 1 na vstupe sa zapíše aktuálny čas. Potom sa sa len čaká, kedy bude prečítaná logická 0, kedy si opäť zapíšeme čas. Tieto dve zapísané hodnoty sú následne dosadené do vzorca na výpočet vzdialenosti.

Táto časť výpisu však nieje kompletná. Táto funkcia je rozšírená o celý IF blok, ktorý kontroluje druhý senzor. Vo výpise je viditeľná len kontrola pre pin 9. Takisto je v tejto časti len jeden zápis na LED, pomocou funkcie writePin.

Funkcie writePin a readPin boli vytvorené za účelom zjednodušenia práce pri písaní, takže momentálne stačí do funkcie posielať len pin, na ktorom chceme čítať alebo zapisovať. Samozrejme pri zapisovaní je funkcia rozšírená aj o atribút logickej hodnoty, ktorú chceme na pin zapísať.

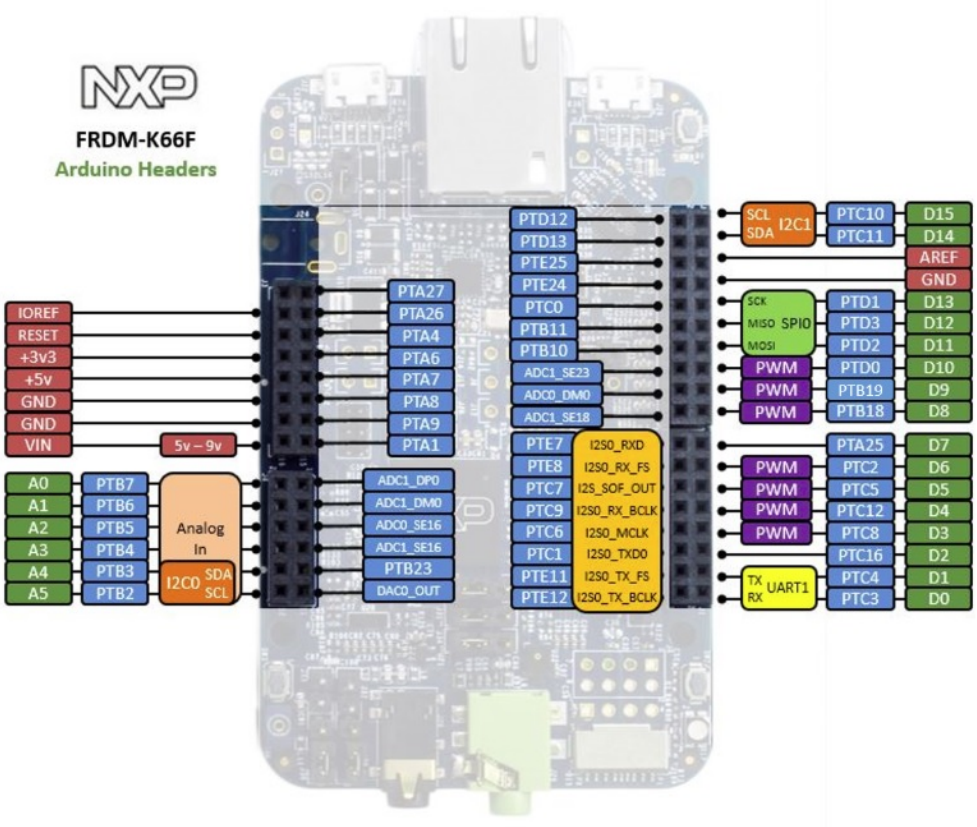
```
extern "C" {

   void PIT0_IRQHandler(void){
       PIT_ClearStatusFlags(PIT, kPIT_Chnl_0, kPIT_TimerFlag);
      time_us+=10;
       __DSB();
   }

   void PORTA_IRQHandler(void)
   {
      uint32_t flags = GPIO_PortGetInterruptFlags(GPIOA);
      if((flags & (1U << 9)) > 0)
      {
          GPIO_PortClearInterruptFlags(GPIOA, 1U << 9);
         if(readPin(9) == 1){
            first_time_start = time_us;
         } else {

            first_time_end = time_us;
            first_dist_irq = (first_time_end - first_time_start) * 0.034/2;

            if(first_dist_irq < max_distance){
               writePin(4, 1);
            } else writePin(4, 0);

            array_first[array_first_tmp] = first_dist_irq;
            array_first_tmp = (array_first_tmp + 1) % 10;
         }
      }
      __DSB();
   }
}
```

Výpis 1: Algoritmus na meranie vzdialenosti pomocou PIT

5 Domáci model auta

Z dôvodu celosvetovo rozšírenej nákazy COVID-19 a uzavretých hraníc som sa nemohol dostaviť do laboratória, aby som mohol pracovať na skutočnom modeli autíčka. Z tohto dôvodu som vytvoril domácu verziu autíčka, na ktorej by som mohol testovať moje zadanie práce.

Základom boli dva ultrazvukové senzory HC-SRO4, pomocou ktorých je možne vysielať ultrazvukové vlny z trazmitora a následne prijímať odrazené vlny na prijímači. Bližšej špecifikácií tohto senzora sa venuje kapitola 4.2. Model autíčka bol vytvorený z obyčajnej krabice, ktorá sa dá nájsť v hociktorej domácnosti. Na tejto krabici sú uložené dve dosky. Jednou je vyššie opísaná FRDM-K66F, ktorá je zapojená cez napájací kábel USB do notebooku. Na tejto doske sa využívajú GPIO piny na napájanie, uzemnenie a šesť programovateľných GPIOA pinov. Celkový vývoj pinov na tejto doske je zobrazený na obrázku 13.



Obr. 13: Pinout dosky FRDM-K66F [5]

Ako je vidieť zo schémy dosky FRDM-K66F, sú tam len 2 piny na uzemnenie a 2 piny napájania, pričom jeden pin je 5V a druhý 3,3V. Pri simulácií tejto práce a celého autíčka som však potreboval použiť 2 ultrazvukové senzory a 2 ledky, ktoré by signalizovali reakcie senzorov. V tomto prípade bolo teda nutné použiť vývojový panel, ktorý som používal k testom na Arduino doske. Na túto vývojovú dosku som pomocou káblov prepojil napájanie 5V a uzemnenie. 5V

napájanie bolo použité len na ultrazvukové senzory HC-SRO4. Uzemniť bolo však potreba všetky zariadenia, teda aj LED kontrolky. K ním bolo potrebné dodať aj 220 Ω rezistory, pretože dané LED kontrolky si brali podľa dokumentácie okolo polovice napätie, čo činí približne 2,5V. Zvyšné napätie sa teda "strácalo"v doske, prípadne by ledka dostávala viac napätia ako potrebuje, čo spôsobuje zahrievanie, ktoré môže viesť až k vypáleniu čipu, alebo ledky.

5.1 Výroba auta

Výroba auta prebiehala v dvoch fázach. Prvotný model vznikol z obyčajnej krabice, na ktorú som naukladal vývojové dosky a senzory boli pinovou časťou vsunuté do kartónovej časti krabice. Káble viedli dovnútra krabice a následne opäť von k doskám. Na prvý model to bolo vcelku dostačujúce, no hneď sa ukázali dôvody, prečo nie vyhovujúce.

Ako prvý problém bol, že senzory neboli v rovnakých polohách. To znamená, že ako aj uhol vertikálny, tak aj uhol horizontálny neboli zhodné. Z tohoto dôvodu bolo potrebné vymyslieť vylepšený model, ktorý by čiastočne tieto problémy riešil a udržal komponenty pevne na autíčku, respektíve na krabici.

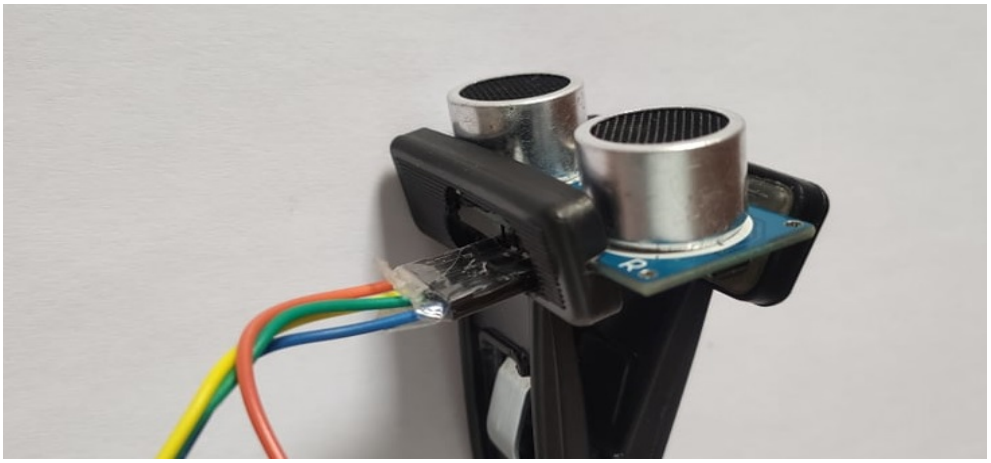
Na návrh nového modelu síce bola použitá nová krabica, tento krát však bola spevnená vrstvou tvrdeného papiera. Zo všetkých strán bola krabica obalená kvôli pevnejšiemu telu, ale aj kvôli neželanej reklame pre jednu nemenovanú značku. Po bokoch autíčka boli nalepené dizajnové kolieska z papiera.

Na uchytenie senzorov som hľadal vhodné uchytenia, ktoré by mi dovoľovali v rámci možnosti nastavovať horizontálny a vertikálny uhol uchytenia senzorov. Toto je veľmi dôležité, pretože senzory sú vcelku citlivé a aj najmenšia zmena uhla môže vytvárať nepresnosti v meraní. Po vyskúšaní rôznych uchytení, som sa uchýlil k možnosti uchytenia pomocou držiakov, ktoré sa dajú nájsť na bežných vešiakoch na nohavice. Na tomto držiaku sú dve kliešťovité uchytenia, do ktorých sa senzor akurát zmestí. Ako prvou úlohou bolo upraviť tieto držiaky tak, aby som mohol senzor uchytiť vo vodorovnej polohe bez toho, aby som nejak poškodil piny na senzore. Z toho dôvodu bolo nutné prevŕtať jednu stranu kliešťovitého uchytenia, aby bolo možné cezeň vsunúť piny, ale stále by bolo uchytenie senzorov nepoškodené.

Senzor v uchytení drží pomerne dobre. Uchytenie je pevné, z vrchnej časti je pogumovaná časť, ktorá ma v sebe horizontálne drážky, vďaka ktorým je umožnené nastavovať z malej časti vertikálny uhol snímača. Problém však nastáva, ak sa s uchytením príliš manipuluje, kedy dochádza k stláčaniu uchopenia, kedy vrchná pogumovaná časť tlačí na hrany senzorov a vytláča ich smerom dopredu pričom spodná časť je tlačená opačným smerom a spodné uchytenie začína tlačiť na pinovú časť, kde môže dôjsť k ohnutiu.

5.1.1 Uchytenie senzorových držiakov

Aj keď sa problém s uchytením senzorov zdal byť vyriešený, komponenty použité na uchytenie senzorov neboli práve vyrobené na mieru pomocou 3D tlačiarne, čo by sa za normálnych okolností



Obr. 14: Uchytenie senzoru

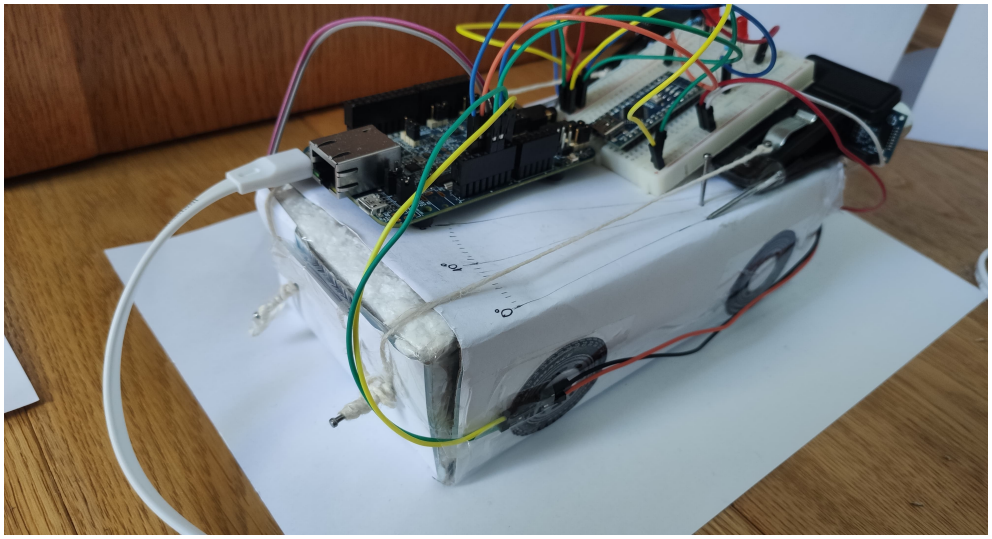
urobilo. Keďže sa stále bavíme len o uchytení z klasického vešiaka, bolo potrebné vymyslieť, ako toto uchytenie zakomponujem do modelu auta. Jediným riešením bolo umiestniť uchytenie tak, aby sa plochou časťou mohol položiť na model auta. V uchytení bol už spravený výrez, ktorý mi dovoľoval dovnútra vložiť takzvanú osku, ktorá v skutočnosti nie je žiadna oska, ale obyčajný klinec. Tento klinec mi dovoľoval pohyb uchytenia v horizontálnom smere.

Tu prišiel nápad s tým, ako pri simuláciách vlastne zobrazovať náklony senzorov, aby sme vedeli zabezpečiť, že meriame v rovnakom náklone s oboma senzormi súčasne.

Senzory boli umiestnené na kraje modelu autíčka, aby bolo možné zabezpečiť vyhýbanie sa prekážkam do strán. Idea je taká, že ak sa prekážka nachádza povedzme v polke modelu autíčka a reálne je možnosť obísť ju tak, aby model nevybehol z dráhy, tak by sa tak stalo. Týmto sa dá zabezpečiť, že túto prekážku by zaznamenal iba senzor z tej strany, z ktorej sa nachádza prekážka.

Na kraje modelu autíčka boli nakreslené zobrazovacie uhly, vďaka ktorým by sa dalo v rámci malej odchýlky zobraziť, pod akým stupňom náklonu je senzor natočený. Na to aby sme mohli na tieto uhly zobrazovať nejakú hodnotu, bol použitý ďalší klinec, ktorý sa prilepil o uchytenie a fungoval ako ručičky na pomyselných hodinách, teda v tomto prípade ukazoval stupne náklonu. Ako som už spomínal, tieto náklony nie sú úplne presné, pretože k dokonalosti chýba lepšie zakomponovanie uchytenia ku kostre autíčka. To by umožňovalo dokonalý pohyb iba v horizontálnej časti, na rozdiel od momentálneho riešenia, ktoré tento pohyb síce umožňuje, ale taktiež je možné celé uchytenie posúvať do strán, čím nie je možné úplne zaistenie presnosti momentálneho uhla natočenia senzoru.

Bližší pohľad na uchytenie je zobrazené na obrázku č.15. Z obrázku je viditeľné, že v časti, kde sa nachádza klinec, ktorý dovoľuje horizontálny pohyb, je pomerne veľký priestor na pohyb do strán. Preto bolo nutné dávať pozor, či nebolo uchytenie senzorov posunuté do strany.



Obr. 15: Riešenie uchytenia

5.1.2 Zaistenie a vertikálna manipulácia uchytenia

Ďalej je na obrázku pozorovateľné akési uchytenie šnúrou, respektíve nitkou. Toto uchytenie slúži najmä na to, aby senzory nepadali pred autíčko. Ako som už vyššie spomínal, v uchytení na senzor boli vyvŕtané otvory pre piny, do ktorých sú následne nasunuté kábliky. Práve tieto piny s káblikmi zabraňujú spätnému pohybu dozadu, ak by sa nitka viac pritiahla, pretože toto zapojenie je zapreté o predné hrany modelu auta. V tomto prípade sa teda môže uchytenie pohybovať len v horizontálnom smere a vertikálny uhol náklonu sa da mierne doladiť pritiahnutím nitky, ktorá je upevnená v zadnej časti o ďalší klinec, na ktorý je namotaná a uviazaná, takže v skutočnosti funguje ako akási uťahovacia skrutka. Klinec je následne zapichnutý do modelu auta, ktorý je zo zadnej strany ešte spevneným kusom polystyrénu, ktorý je nalepený na zadnej strane modelu autíčka. Táto zadná strana je taktiež aj otvárateľná tak, že takzvané pánty sú na spodnej strane krabice, čo zabezpečuje otváranie zadnej časti smerom dole a teda aj dodatočné napnutie nitky v prípade, že by mechanizmus uchytenia takzvanej uťahovacej skrutky zlyhal, čo sa môže ľahko stať, nakoľko klinec nemá vo svojej konštrukcií žiadne drážky, ktoré by zabraňovali uvoľňovaniu napätia a teda spätnému otáčaniu, ktoré by viedlo k znižovaniu horizontálnemu náklonu senzoru.

5.2 Výroba pomôcok na simulácie

K tomu aby sme mohli vlastne otestovať vyrobený model auta, bolo nutné taktiež vytvoriť časť dráhy a prekážku, na ktorej by sa dalo testovať fungovanie senzorov.

5.2.1 Dráha na testovanie presnosti

Táto dráha je vytvorená z dvoch papierov 4A, ktoré su zlepené po výške. Na túto dráhu boli namerané vzdialenosti po 5cm, čo vytvorilo 10 dielikov s celkovou vzdialenosťou 50cm, na ktorej bude možné otestovať presnosť merania ultrazvukových senzorov.

5.2.2 Prekážka

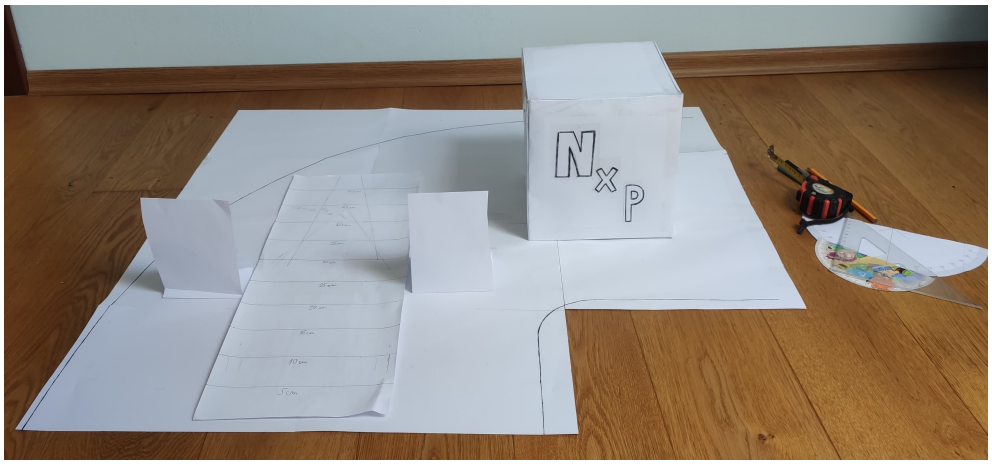
Jedna z disciplín NXP Cupu je vyhnutie sa prekážke s rozmermi 20x20x20cm. Na výrobu tejto kocky bolo potreba zopár papierov a trochu trpezlivosti. Kocku sa podarilo vytvoriť v pomerne presných rozmeroch, aj keď niektoré strany sú o pár milimetrov kratšie, na domáce simulácie to však bude postačujúce.

5.2.3 Testovacia dráha

Aby bola simulácia čo najreálnejšia, vytvoril som testovaciu dráhu podľa pravidiel NXP Cupu. Nevytvárala sa celá dráha, ale len jej časť s 90° zatáčkou. Šírka testovacej dráhy je 55cm a je vyrobená z klasického A4 papiera, ktorý bol zlepený aby vytvoril dostatočne veľkú plochu. Na vytvorenie zatáčky bolo potrebné vytvoriť domáce kružidlo, ktoré bolo vytvorené s pomocou klasického stavbárskeho metra, na ktorom bola nastavená hodnota na 55cm.Na konci metra bola upevnená písacia časť, ktorú predstavovala obyčajná ceruzka. Ako prvá bola spravená vnútorná čiara za pomoci obkreslenia zaoblenej časti uhlomera. Následne za pomoci tejto vnútornej čiary a domáceho kružidla bolo možné vytvoriť vonkajšiu čiaru.

5.2.4 Prekážky na meranie uhlov

Ako posledná pomôcka, respektíve pomôcky boli vytvorené papierové prekážky, ktoré sú široké približne 10cm, ktoré sa budú ukladať pred senzory s rôznym rozostupom, vďaka čomu bude možné odmerať uhol záberu senzorov.



Obr. 16: Pomôcky na simulácie

6 Simulácie a experimenty s ultrazvukovými senzormi

Aby sme vedeli správne posúdiť presnosť meraní, bolo treba najprv skontrolovať, či sú senzory v približne, ak nie presne na rovnakej pozícii. Na kalibrovanie uchytenia senzorov najprv používal knihu, alebo hociktorý rovný predmet, ktorý sa dal priložiť k senzorom tak, aby bolo vizuálne vidieť , či sú zarovnané rovno, alebo pod nejakým uhlom. Ako je už vyššie spomenuté, nastavenia pozícií senzorov nebolo vždy perfektné a preto nasledujúce simulácie bude treba brať s rezervou.

6.1 Testovanie presnosti merania

Na otestovanie presnosti merania senzorov bola použitá dráha na testovanie presnosti, ktorou je možné merať do vzdialenosti pol metra. Testy prebiehali pri rôznych vzdialenostiach, ale aj náklonoch senzorov. Jedným zo spôsobov kontroly vzdialenosti je výpis do konzoly. Tento spôsob sa však neosvedčil, pretože výpis do konzole s najväčšou pravdepodobnosťou používa tiež nejaké prerušenie na časovanie komunikacie cez USB, čo zapríčiňovalo zlý výpis hodnôt. Táto chyba sa dala eliminovať tým, že sa vypisovalo len v čase každej 10 nameranej hodnoty. To znamená, že výpis hodnôt sa najskôr zapísal do statického poľa o veľkosti 10 a po jeho naplnení sa jeho hodnoty vypísali, čo vylepšilo štatistiku presnosti výpisu hodnôt na 9/10.

Meranie nakoniec prebiehalo pomocou LED kontroliek, ktoré detekovali najspoľahlivejšie aktuálne vzdialenosti objektov. Je to najmä jednoduchšie, pretože stačí nastaviť globálnu premennú na určitú vzdialenosť a následne model autíčka priložiť do takej vzdialenosti, akú sme si zvolili a sledovať LED kontrolky. Tieto testy sa robili pri rôznych vzdialenostiach a uhloch natočenia senzorov. Celkové výsledky meraní sú zobrazené v tabuľke 2.

Tabuľka 3: Výsledky testovania senzorov

Meraná vzdialenosť	Skutočná vzdialenosť pri nameraní	Uhol senzorov
20cm	20,4cm	0°
30cm	30,4cm	0°
40cm	40,5cm	0°
50cm	50,5cm	0°

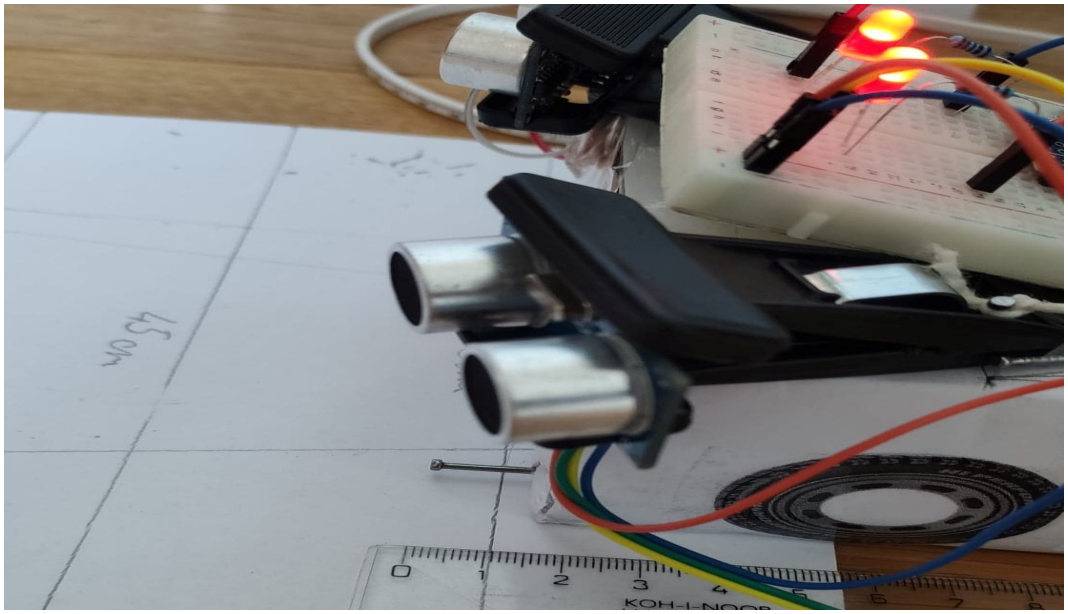
Z tabuľky je zrejmé, že odchýlka nameraných vzdialeností sa líši od skutočnosti len o 0,5cm v priemere. Všetky tieto hodnoty sú zaokrúhlené, nakoľko meranie prebiehalo klasickým stavbárskym metrom a desatinné hodnoty boli niekedy hraničné. Testy následne prebiehali pod uhlom 10° pre každý senzor, kedy boli hodnoty približne rovnaké a väčšie výkyvy boli zaznamenané až pri uhloch okolo 20°Ḣodnoty z meraní sú popísané v tabuľke 3.

Zvýšene nepresnosti meraní by sa dali pripísať k rozdielnym polohám prijímača a vysielača, kde pri 20° uhle bola 1 z komponent senzora o približne 1cm bližšie k prekážke ako druhá komponenta V tomto prípade na ľavom senzore je prijímač o 1 cm bližšie ku konštrukcií modelu autíčka ako vysielač. Naopak na pravom senzore je o 1cm bližšie ku konštrukcií vysielač. Táto

Tabuľka 4: Výsledky testovania senzorov

Meraná vzdialenosť	Skutočná vzdialenosť pri nameraní	Uhol senzorov
20cm	21cm	20°
30cm	31cm	20°
40cm	41,2cm	20°
50cm	41,3cm	20°

teória sa však dá vyvrátiť tvrdením, že zvuk ma konštantnú rýchlosť a vzdialenosť vysielač - prekážka - prijímač by mala byť rovnaká, nakoľko v prípade ľavého senzora je vysielač bližšie k prekážke, ale o to má dlhšiu cestu k prijímaču. Keďže my potrebujeme sledovať celkový čas a nie čas medzi komponentou a prekážkou, táto teória nemá pevné základy, tým pádom nie je úplne jasné, čo môže nepresnosti spôsobovať.



Obr. 17: Náklon snímačov a ich rozpoloženie

6.2 Kužeľ snímania

Na obrázku 17. sa dá všimnúť, že okrem čiar vzdialenosti sa tam nachádzajú aj šikmé čiary. Tieto čiary vznikali pri meraní uhlu snímania, alebo inak povedané, v tomto teste budem sledovať "kužeľ snímania".

Podľa dokumentácie senzoru HC-SRO4, by mal byť uhol snímania jedného senzoru 15°, čo sa meraním prekvapivo potvrdilo a čím som si potvrdil správny spôsob merania uhlov. Meranie spočívalo na jednoduchom princípe. Model autíčka bol rovnako, ako na predošlej simulácií položené na dráhe testovania presnosti. Model auta sa obkreslil, aby bola zaistená rovnaká pozícia

modelu autíčka, pri každom teste. Merateľná vzdialenosť, kedy sa majú LED kontrolky rozsvietiť nastavená na 25 cm. Model autíčka bol umiestnený vo vzdialenosti 30cm oproti prekážke, čo značilo ledky vo vypnutom stave. Postupne pri rôznych uhloch náklonu boli pridávané prekážky na meranie uhlov. Prekážky sa pridávali z vonkajšej strany vo vzdialenosti asi 20 cm od modelu autíčka. Postupne sa prekážky posúvali bližšie k sebe, až, kým kontrolky LED nezačali preblikávať, alebo úplne svietiť. Uhol snímania sa testoval pri rôznych uhloch senzorov od ich nulovej polohy. Namerané hodnoty je možné vidieť v tabuľke 4.

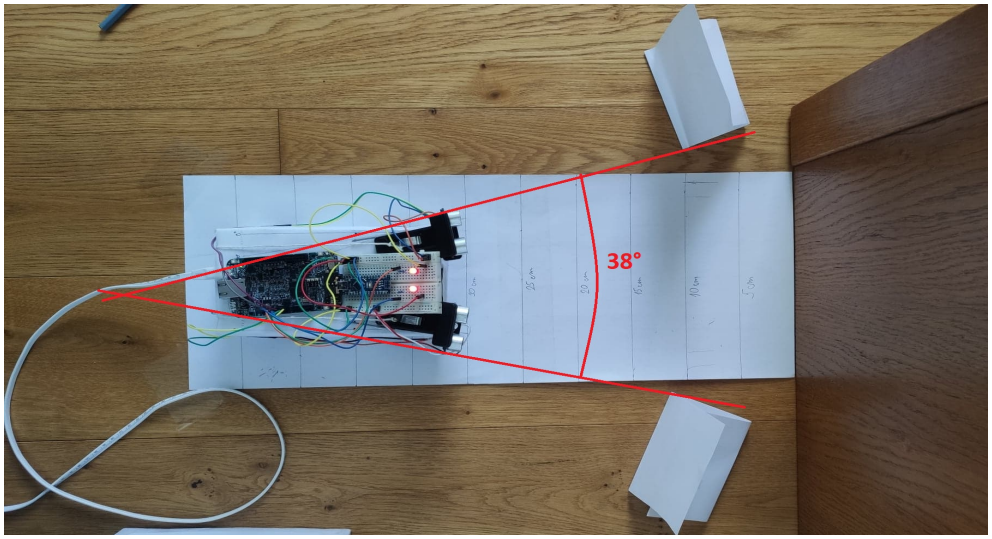
Tabuľka 5: Výsledky testovania senzorov pri rôznych uhloch náklonu

Náklon senzorov od nulovej polohy	Uhol snímania
0°	15°
5°	21°
10°	38°
15°	50°
20°	60°

Spôsob merania uhlov spočíval v tom, že po nájdení polôh pre testovacie prekážky sa tieto polohy zakreslili a následne sa nakreslili dlhé čiary na testovacie dráhy, ktoré sa v určitom momente prekrížili. Po nájdení bodu, kde sa tieto čiary prekrížili, bolo možné odmerať približný uhol snímania senzorov. Pri testovaní som si taktiež všimol, že objekt bol nasnímaný takmer vždy, keď bola prekážka rovnobežne so senzorom ktorý ju snímal a stačilo, ak sa táto prekážka dostala hranou do približne stredu dosky senzora. Na súťaži NXP Cup však budú prekážky položené tak, že bočné hrany kocky budú rovnobežné s bočnými čiarami trate. Tu môže nastať situácia, kedy pri približne 15-20° uhle náklonu senzorov dochádza k javu, kedy je senzor 'slepý' aj keď je práve namierený oproti prekážke. Tento jav je spomenutý v podkapitole 6.3.4. Treba len pripomenúť, že výsledky treba opäť brať s rezervou, nakoľko uhly senzorov nemuseli byť nastavené presne, vzhľadom na konštrukčné nedostatky popísané vyššie. Presný postup merania je zobrazený na obrázku č.18.

6.3 Simulácia jazdy s kockou

Táto simulácia je pre túto prácu kľúčová, pretože disciplína na vyhnutie sa prekážke je vlastne hlavný dôvod, prečo boli na model autíčka umiestnené ultrazvukové senzory. Pri simulácií bolo potrebné zvážiť všetky možné situácie, ktoré by mohli nastať počas jazdy po trati. Ako prvé som sa snažil nájsť správny náklon snímačov tak, aby pri jazde dochádzalo čo k najmenšiemu počtu situácií, ktoré by viedli k núdzovému brzdeniu. V nasledujúcich pod sekciách sa budem venovať rozboru správania sa modelu autíčka, pri rôznych náklonoch senzorov. Tieto simulácie sa odvíjajú od predpokladu, že autíčko vo svojom základnom režime jazdí na rovnej trati približne v strede jazdnej dráhy a v zatáčkach si nabieha k vnútornej hrane, respektíve čiary.



Obr. 18: Ukážka spôsobu merania uhlov snímania

6.3.1 Nulové polohy senzorov

Ako prvé som testoval nastavenie, kedy boli oba snímače nastavené na uhol náklonu 0°. Pri tomto nastavení autíčko jazdí spoľahlivo a ako je spomenuté vyššie, presnosť merania je najpresnejšia. Táto vlastnosť je však v tejto disciplíne povedzme nepodstatná, nakoľko autíčko musí vedieť reagovať na vzniknutú situáciu včas a odchýlka presnosti merania na hodnotách 1cm sú zanedbateľné.

Autíčko spoľahlivo detekuje objekty či už pri výjazde zo zatáčky, alebo na rovnej časti trati. Najväčším nedostatkom tohto nastavenia je fakt, že pri nulovom náklone senzorov je veľmi ľahké prísť k situácií, kedy sa rozsvietia obe LED kontrolky. Ak by sme sa držali vyššie spomenutej teórie o jazde autíčka, tak by k núdzovému brzdeniu dochádzalo pomerne často. Prekážka ma totiž šírku 20cm, čo je viac ako jednu tretinu šírky jazdnej trate. Ak by teda model autíčka v rovnej časti trati išiel približne stredom, k núdzovému brzdeniu by dochádzalo takmer vždy pri strete s prekážkou. Ak by bola prekážka posunutá viac k stredu, trate, bolo by takmer nemožné riešiť túto situáciu. K riešeniu rôznych situácií, ako by mohlo autíčko obchádzať prekážky, sa venujem podrobnejšie v kapitole 6.4.

6.3.2 Test s náklonom senzorov do 10°

Pri tomto testovaní bolo ihneď viditeľné, že uhol snímania veľmi rozhoduje, ako sa bude dať vyhýbať prekážkam. Naproti nastaveniu senzorov s nulovými polohami, je toto nastavenie oveľa flexibilnejšie. Za predpokladu, že autíčko ide stredom jazdnej dráhy, nastáva počet situácií s núdzovým brzdením viditeľne menej a modelu autíčka neprekáža, respektíve kocka so svojou hranou takmer v strede jazdnej dráhy a aj tak dochádzalo k rozsvieteniu oboch LED kontroliek, signalizujúcich núdzové brzdenie len občas.

6.3.3 Test s náklonom senzorov do 20°

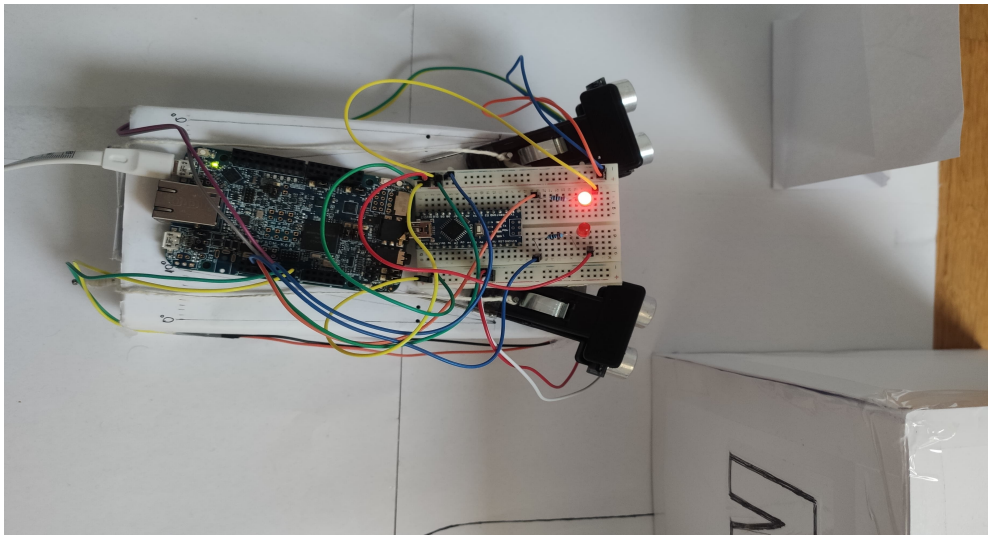
Táto simulácia prebiehala poväčšine s uhlom natočenia senzorov do 20°, nakoľko pri tomto uhle sú už senzory príliš vychýlené. Podľa môjho názoru je ideálny náklon okolo 15°, čo ale najmä závisí od šírky modelu autíčka. Mnou vyrobený model má šírku 10cm, pričom vzdialenosť stredov senzorových dosiek bola medzi 7 - 8 cm. Avšak šírka reálneho modelu autíčka je o 6 cm väčšia, čo dovoľuje osadiť snímače o niečo ďalej od seba, čo môže spôsobiť, že ideálny náklon senzorov bude podľa môjho skromného názoru na hodnote približne 10°.

Čo sa týka simulácie, tu boli výsledky o niečo lepšie ako pri náklone 10°, čo hlavne dovoľovalo prekážkam byť čo najviac vsunuté do dráhy. Podľa pravidiel NXP Cupu by mala byť prekážka položená tak, aby sa dala obísť. V tomto prípade je nastavenia uhla senzorov najflexibilnejšie k nastanej situácií a teda by bolo možné obchádzať prekážky aj pri hraničných situáciách, bez toho, aby sme sa dostali do núdzového brzdenia. Pri tomto náklone sa však začínal viac objavovať zvláštny jav, ktorý som nazval 'Slepý senzor' a je popísaný v podkapitole nižšie.

6.3.4 Slepý senzor

Tento jav nastával pri simuláciách, kde som sa snažil zistiť ideálny uhol náklonu senzorov, ktorý by bol najvhodnejší na vyhýbanie sa prekážkam na trati. Ako som vyššie spomínal v kapitole 6.2, senzor dokáže nasnímať objekt ktorý je rovnobežný so senzorom , ak je jeho hrana približne v strede senzora, respektíve je hrana v polovici dĺžky dosky senzora. Takáto situácia však nebude nastávať vždy, skôr bude výnimočná. Najčastejšie sa budeme stretávať so situáciou, kedy senzor a prekážka nebudú na seba rovnobežné a teda zvukové vlny sa nebudú odrážať v najideálnejších uhloch. Pri testovaní som si všimol momentu, kedy bol model aut položený na dráhe a postupne som na jeden zo senzorov zo strany nastavil prekážku z vonkajšej strany tak, aby bola na hraničnej pozícií. Presnejšie povedané, kontrolka LED, ktorá je priradená k danému senzoru začala preblikávať, pretože dostávala protichodné hodnoty, pretože objekt bol na hranici sponzorovateľnosti. Pri ďalšom posunutí prekážky o pár milimetrov do uhlu snímania sa ledka upokojila a ustálila na čistom rozsvietení, čim indikovala prekážku. Pri ďalšom posunutí prekážky o pár milimetrov do uhla snímania, sa kontrolka vypla napriek tomu, že senzor bol priamo namierený na prekážku. Pri ďalšom posunutí sa ledka opäť rozsvietila.

Jedným z možných vysvetlení môže byť to, že v niektorých situáciách je uhol vysielania a následného odrazu taký, že daný signál sa odráža opačným smerom, ako by sme chceli a tým senzor neprijíma žiadne informácie o objekte pred ním. To je však ničím nepodložené tvrdenie, nakoľko vizualizácia šírenia ultrazvukových vĺn je o niečo komplikovanejšia, ako vizualizácia odrazu svetelných lúčov. Je nutné poznamenať, že tento jav som si všimol len vtedy, keď bol model autíčka na rovnakej pozícií a prekážka sa objavovala postupne v hraničnej zóne snímania senzoru. Taktiež stojí za povšimnutie, že tento jav sa začal objavovať až pri veľkom náklone snímačov, kedy uhol náklonu dosahoval okolo 20°.



Obr. 19: Ukážka slepého senzoru

6.4 Možnosti riešenia jazdy

V tejto časti by som chcel popísať približne moju ideu toho, ako by mohlo byť autíčko naprogramované tak, aby bolo schopné vyhýbať sa prekážkam čo najefektívnejšie. Táto časť bude trošku viac abstraktná, nakoľko je to len moja idea, ktorá neje momentálne nijak otestovaná a z pohodlia domova je ťažké predpovedať, ako sa bude model autíčka skutočne správať po implementácií ultrazvukových senzorov.

Ako je spomenuté vyššie, ideálny náklon podľa môjho názoru, vzhľadom na šírku modelu autíčka by bol približne 10°. Je to zlatá stredná cesta, čo sa týka všetkých vyššie spomenutých testov, nakoľko sa nestretávame často so situáciami, vyžadujúce núdzové brzdenie, pokiaľ nejde o prekážku, ktorá je položená cez celú testovaciu dráhu. Taktiež pri tomto uhle boli minimálne pozorovateľné situácie, kedy bol senzor "slepý". Aj keď tento jav sa vyskytoval len pri testoch, kedy sa modelom autíčka nehýbalo, neznamená to, že takáto situácia nemôže nastať a treba s ňou rátať.

Jediný problém môže nastať, ak by prekážka bola postavená blízko zatáčky na trati, čo by mohlo v niektorých situáciách spôsobiť, že autíčko môže nasnímať prekážku oboma senzormi. Táto situácia nenastávala nejak často a závisia hlavne na polohe kocky ale aj od toho, ako ďaleko prechádzalo autíčko od vnútornej čiary zatáčky a na presné zistenie situácie by som potreboval urobiť testy priamo v školskom laboratóriu na skutočnom modeli autíčka.

Čo sa týka mojej idei o tom , ako by sa mohlo autíčko efektívne vyhýbať prekážkam, tak by bolo nutné nastaviť nejaké zóny vzdialeností, ktoré by sa ignorovali a ktoré by sa dali špecificky riešiť. Zóny by boli rozdelené tak, aby každá zóna predstavovala iné natočenie kolies, čím bude možné postupné vyhnutie sa prekážke namiesto prudkého vybočenia, ktoré bude môcť vyústiť do ďalších nepredvídateľných situácii.

Ako príklad by sa dal uviesť klasická situácia, kedy sa autíčko rúti po rovnej dráhe a prekážka by bola v umiestnená na kraji dráhy z jednej zo strán. V tomto momente by mal reagovať iba senzor na tej strane, na ktorej sa nachádza prekážka. Vo chvíli kedy by senzor spozoroval prekážku by sa nachádzal v prvej zóne, kedy by začalo autíčko postupne jemne zatáčať a vyhýbať sa prekážke. Najideálnejšie by bolo, keby sa dokázalo autíčko vyhnúť prekážke už v prvej, prinajhoršom v druhej zóne. V momente, kedy by už senzor nebol schopný zaznamenať danú prekážku v ceste, by sa kolesa vyrovnali, čo však môže spôsobiť rôzne problémy.

Jeden z problémov je ten, že po tom ako autíčko prestane zatáčať a vyrovná natočenie kolies, autíčko pôjde rovno, ale nie rovno po trati. Pôjde rovno pod takým uhlom, pod akým končilo vyhýbanie sa prekážke. Tu môžu nastať problémy, ktoré sa neodvažujem ani odhadnúť, ako môže reálny model autíčka zareagovať vzhľadom na momentálne nastavenia a algoritmy. Jasné je len to, že je nutné vyrovnať smer jazdy tak, aby išlo rovno a čo najviac v strede jazdnej trate. Tu môže nastať ďalší problém s tým, kedy začať vyrovnávať smer jazdy. To môže byť relatívne, a ideálne by sa malo nejak prepočítavať, pretože ak by sa ihneď po skončení vyhýbania sa začalo autíčko vyrovnávať, môže veľmi ľahko nastáť situácia, kedy autíčko môže prechádzať okolo prekážky vo veľmi blízkej vzdialenosti, ba až sa môže do nej jemne naraziť bez toho, aby snímače niečo spozorovali.

6.5 Test núdzového brzdenia

Test núdzového brzdenia prebiehal v podstate rovnako, ako test presnosti merania, pretože namerané hodnoty a ich odchýlky sa vyvíjajú od stavu, kedy svietia obidve LED kontrolky naraz. Pri domácom testovaní to bol asi jediný možný spôsob, kedy by mal model autíčka núdzovo brzdiť, nakoľko podľa pravidiel z NXP Cupu, by mala byť prekážka cez trať položená na rovnej časti trate. Na to, aby zasvietili obe LED kontrolky v tomto prípade je irelevantné horizontálne nastavenie uhlov senzorov, nakoľko senzory aj pri najväčších testovaných uhloch nahnutia senzorov dokázali včas eliminovať prekážku. V tomto prípade v podstate neexistuje nejaký dôvod , prečo by nemalo byť núdzové brzdenie vyvolané, pretože návrh osadenia a simulovania senzorov bolo zamerané na to, aby sme sa čo najviac vyhýbali situáciám s núdzovým brzdením a ak už núdzové brzdenie bolo vyvolané, tak len vo veľmi špecifických prípadoch.

7 Záver

Cieľom tejto práce bolo vytvorenie simulačného modelu autíčka, ktorý by bol schopný čo najlepšie simulovať situácie, ktoré môžu nastať pri skutočných testoch na reálnom modeli autíčka. Na tento model boli použité dva ultrazvukové senzory, ktoré boli osadené v rôznych uhloch a dokázali spoľahlivo detekovať prekážky, objavujúce pred nimi. Počas práce boli nasimulované všetky možné situácie, ktoré z mojho pohľadu mohli nastať. Samozrejme všetko treba brať s rezervou, pretože moje predpoklady môžu byť správne, ale taktiež nemusia a to či daný model bude môcť jazdiť podľa mojích predstáv, by bolo potreba vyskúšať v školskom laboratóriu na skutočnom modeli autíčka.

Najdôležitejšie bolo vytvorenie fungujúceho algoritmu, ktorým by bolo možné s pomerne veľkou presnosťou možné merať vzdialenosti objektov pred senzormi. Následne bolo potrebné sa zamerať na funkčnú časť a stavbu domáceho modelu autíčka tak, aby bolo možne čo najpresnejšie nasimulovať situácie, ktoré by boli podobné tím, ktoré by sa objavovali pri testoch v školskom laboratóriu.

Táto práca opisuje výrobu modelu autíčka a všetky technické riešenia, ktoré boli v domácich podmienkach prístupné, čo znamená, že pri vývoji bolo potrebné mať trochu predstavivosti a fantázie.

Na vytvorenie simulácie bolo treba taktiež vytvoriť nejaké podporné materiály, s ktorými by sa dala otestovať skutočná funkčnosť vyrobeného modelu. Na záver simulácií som sa snažil vždy vyjadriť k problémom, ktoré by mohli teoreticky nastať a touto cestou aj vyjadriť svoj osobný názor a ideu možného fungovania v budúcnosti.

Literatúra

1. *NXP CUP RULES 2019/20: Arduino and NXP Header Pinout.* Dostupné tiež z: https://community.nxp.com/docs/DOC-335269.
2. *NXP Cup: Students Build and Race Autonomous Model Cars.* Dostupné tiež z: https://blog.nxp.com/automotive/nxp-cup-students-build-and-race-autonomous-model-cars.
3. ZVONEK, Richard. *Mechanizmy řízení robotického auta NXP (FREESCALE): Product features* [online] [cit. 2019]. Dostupné z: https://dspace5.vsb.cz/bitstream/handle/10084/136273/ZVO0016_FEI_B2647_2612R025_2019.pdf?sequence=1&isAllowed=y.
4. *The new Alamak NXP Cup Car Kit* [online]. 2006 [cit. 2017-12-01]. Dostupné z: https://community.nxp.com/thread/462715.
5. *FRDM-K66F: Arduino and NXP Header Pinout* [online] [cit. 2020]. Dostupné z: https://os.mbed.com/platforms/FRDM-K66F/.
6. *FRDM-KL25Z: Freedom Development Platform for Kinetis® KL14, KL15, KL24, KL25 MCUs: Specifications.* Dostupné tiež z: https://www.nxp.com/design/development-boards/freedom-development-boards/mcu-boards/freedom-development-platform-for-kinetis-kl14-kl15-kl24-kl25-mcus:FRDM-KL25Z.
7. *Raspberry Pi 3 Model B: Specification.* Dostupné tiež z: https://www.raspberrypi.org/products/raspberry-pi-3-model-b/.
8. *Black or white. The infrared sensor. What is an infrared sensor?* [online] [cit. 2015-10-27]. Dostupné z: http://diwo.bq.com/en/black-or-white-the-infrared-sensor/.
9. *Mediánový filter* [online] [cit. 2016-12-01]. Dostupné z: http://ibooks.sk/publ/12pavlovicova/pdf/kap45.pdf.
10. RAFAEL C. GONZALEZ, Richard E. Woods. *Digital Image Processing.* Prentice Hall, 2008.
11. *A straightforward introduction to Image Thresholding using python: What is Thresholding?* [online] [cit. 2019-10-02]. Dostupné z: https://medium.com/spinor/a-straightforward-introduction-to-image-thresholding-using-python-f1c085f02d5e.
12. *NXP Freescale Cup UK Qualifier at Imperial* [online] [cit. 2016]. Dostupné z: https://wp.doc.ic.ac.uk/hipeds/2016/04/12/nxp-freescale-cup-uk-qualifier-at-imperial/.
13. *Periodic Interrupt Timer* [online] [cit. 2016-10-06]. Dostupné z: https://www.nxp.com/video/periodic-interrupt-timer:PERIODIC-INTERRUPT-TIMER.
14. *Ultrasonic Ranging Module HC-SR04: Product features.* Dostupné tiež z: https://cdn.sparkfun.com/datasheets/Sensors/Proximity/HCSR04.pdf.